A Gradient Perspective on RLVR Stability and Winner Advantage Policy Optimization

Prasanth YSS* **Zhichen Ren*** **Rasa Hosseinzadeh** **Ilan Gofman** **Yuqi Chen**
Zhaoyan Liu **Guangwei Yu** **Jesse C. Cresswell** **Satya Krishna Gorti**
Layer 6 AI, Toronto, Canada

Abstract

Reinforcement learning with verifiable rewards (RLVR) improves language-model reasoning, but GRPO-style optimization remains prone to collapse. We analyse this instability through token-level gradient dynamics, deriving a taxonomy that predicts how updates affect next-token probabilities and entropy. The taxonomy shows that stability depends jointly on the advantage sign and token distribution under the current policy. Motivated by this finding, we propose Winner Advantage Policy Optimization (WAPO), a simple online clipped policy-gradient objective that updates only on positive-advantage completions. Across mathematical reasoning and multi-hop QA benchmarks, WAPO improves training stability and matches or outperforms baselines across multiple model families. Full code can be found at https://github.com/layer6ai-labs/wapo.

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) (Pyatkin et al., 2025) has become a central recipe for improving language-model reasoning and planning (Wang et al., 2025a; He et al., 2025). GRPO (Shao et al., 2024) and its variants (Yu et al., 2025; Liu et al., 2025; Chen et al., 2025; Zheng et al., 2025b; Deng et al., 2025; Qi et al., 2026) use off-policy samples as introduced by Schulman et al. (2017). Rollouts generated by the old policy model $\pi_{\theta_{\text{old}}}$ are used for several gradient updates of the current policy model π_θ for sample efficiency. However, off-policy optimization methods remain fragile and susceptible to collapse (see Figure 1).

Recent studies of collapse share a premise that collapse is caused by drift between old and new policies, and the mismatch between training and inference engines (Deng et al., 2025; Qi et al., 2026; Zhang et al., 2026). Proposed solutions include importance sampling and trust region clipping (Schulman et al., 2015; Qi et al., 2025; Zheng et al., 2025a,b). These approaches are effective in practice, but do not predictably improve stability. We observe that removing more divergent tokens does not always lead to improved stability, and can cause collapse (see Figure 2).

High-entropy collapse
Math, SmolLM3-3B, step 150

```
<think> ))^x28**({}) orderedList x231(color872426).txt . . .
Japanese/Arabic/CJK fragments . . .  identifier BERKELEY uc
. . .
OS受 . . .  PEER assembly . . .  ★★★★★ . . .  server . . .
```

⇒ output spreads into task-irrelevant multilingual, code-like, and malformed text.

Low-entropy collapse
Math, Qwen-3-4B, step 139

```
<think>
</think>
Answer: <your answer>
1000000000000000000000000000000000000000000000000000000000
. . .
```

⇒ output contracts to a repetitive, malformed answer pattern.

Figure 1: Examples of collapsed generations during RLVR training. High-entropy collapse produces diverse task-irrelevant continuations, while low-entropy collapse degenerates into repetitive malformed text.

We take an alternate approach in this study and do not assume all updates to divergent tokens are harmful. Instead of asking "How far are we from the old policy?", we ask "How does a gradient update alter the local distribution?". Our analysis depends on the probability distribution of the current policy model π_θ and does not require explicitly modeling drift from $\pi_{\theta_{\text{old}}}$. Our approach provides an alternative lens for understanding instability when off-policy updates are unavoidable.

In Section 3 we study the first-order gradient effect of policy updates on probability and entropy. We propose a taxonomy that theoretically predicts the separation between prior-aligned tokens that reduce entropy, and prior-opposed tokens which increase entropy. Exploration-like updates

*Equal contribution. PY: theory and experiments, ZR: engineering and experiments.

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 53806–53826
October 24-29, 2026 ©2026 Association for Computational Linguistics

increase entropy and lead to collapsed generations, while prior-aligning tokens in negative rollouts also causes collapse. While some negative-advantage tokens may contain useful learning signal, selecting those tokens under coarse rewards is difficult and lies outside the scope of the current taxonomy.

Motivated by these observations, we propose *Winner Advantage Policy Optimization* (WAPO), a minimal modification to GRPO-style training. WAPO utilizes only positive-advantage completions in its policy-gradient update. Prompts with at least one winner are normalized over the group, while prompts with no positive-advantage completion contribute no policy-gradient. Unlike rejection fine-tuning or simple filtering, WAPO remains an online clipped policy-gradient method: it uses policy ratios and group-centered advantages, but masks policy-gradient terms from non-winning completions.

We analyse WAPO in an idealized binary-reward setting. For a given prompt x, we show that various choices of normalization over positive rollouts ascend the same success-probability q_x, but with an adaptive factor that depends on q_x. For WAPO, this factor is $1 - q_x$, which reduces update strength as prompts become saturated while preserving reinforcement from sampled successes.

Empirically, we evaluate WAPO comprehensively on NuminaMath-LEAN (Wang et al., 2025a), Math-500 (Lightman et al., 2024), Hotpot-QA (Ho et al., 2020), and OTT-QA (Chen et al., 2021), covering mathematical reasoning and multi-hop QA datasets. We compare across model families including Qwen3-4B (Yang et al., 2025), SmolLM3-3B (Bakouch et al., 2025), and Gemma3-4B (Kamath et al., 2025). Across these settings, WAPO matches or improves over GRPO, DAPO (Yu et al., 2025), and GSPO (Zheng et al., 2025b) in final accuracy and pass@k. The gains are clearest when baselines collapse; when baselines are stable, WAPO performs on par despite using only positive-advantage completions for policy-gradient updates. We also show out-of-distribution generalization of WAPO by evaluating Hotpot-QA and NuminaMath-LEAN checkpoints on 2-wiki (Ho et al., 2020) and AIME'25 (Dekoninck et al., 2026) respectively.

2 Background

RLVR commonly uses GRPO-style objectives, where a rollout policy $\pi_{\theta_{\text{old}}}$ samples grouped completions, rewards are converted into group-relative advantages, and sequence-level advantages are applied through clipped importance-ratio updates, optionally with KL regularization to $\pi_{\theta_{\text{old}}}$ (Shao et al., 2024). Variants such as DAPO (Yu et al., 2025), which modifies length normalization and clipping, and Dr. GRPO (Liu et al., 2025), which analyses reward and length-normalization biases, refine this framework, yet off-policy optimization under coarse rewards remains prone to collapse (Deng et al., 2025; Qi et al., 2026).

Policy mismatch and trust-regions. Instability is most often attributed to mismatch between $\pi_{\theta_{\text{old}}}$ and the current policy π_θ. One line of work refines trust-region mechanisms: GSPO (Zheng et al., 2025b) addresses the use of token-level importance sampling with sequence-level clipping, while Qi et al. (2026) show that ratio-based clipping can amplify harmful gradients on low-probability tokens. Another line targets train-inference mismatch through engineering fixes, including FP16 precision alignment (Qi et al., 2025), adaptive learning-rate scheduling (Zhang et al., 2026), and router synchronization for MoE architectures (Ma et al., 2025; Ye et al., 2026). These approaches all treat policy or system mismatch as central; our results instead motivate analysing collapse through token-level gradient effects.

Gradient analysis of training dynamics. Gradient-based analyses provide a complementary view. Deng et al. (2025) show that failed low-likelihood trajectories can induce likelihood displacement and drive collapse. In DPO-style (Rafailov et al., 2023) off-policy preference optimization, Ren and Sutherland (2025) identify a related "squeezing effect": negative updates reduce the likelihood of all responses, not only penalized ones, yielding a degenerate peaked distribution.

These works motivate our focus on how advantage sign interacts with the current token distribution during GRPO-style training. Our taxonomy in Section 3 unifies positive- and negative-advantage updates under this token-gradient view and motivates the positive-only update used by WAPO.

3 A Gradient-Motivated Token Taxonomy

We train SmolLM3-3B on OTT-QA with DAPO under different clipping thresholds (see Figure 2). Lowering the clipping threshold ϵ_{low} removes divergent updates, but training collapses. Thus, di-

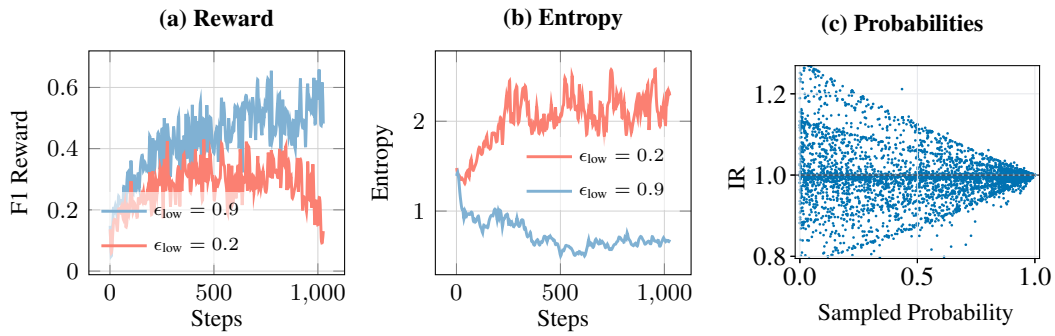


Figure 2: We plot F1 reward (a) and entropy (b) of SmolLM3-3B trained with DAPO under two negative clipping thresholds ϵ_{low} on OTT-QA. More aggressive clipping (red) suppresses divergent negative tokens, but does not improve training stability, nor reduce entropy. Panel (c) shows importance ratio (IR) vs. sampled probability at one of the steps. The spread of importance ratio at lower probability indicates that clipping adversely affects lower-probability tokens (Qi et al., 2026). In Section 3 we show that restricting low-probability negative tokens leads to an increase in entropy and drives collapse.

vergence from $\pi_{\theta_{\text{old}}}$ alone does not predict stability. Figure 2c shows ratio-based clipping adversely affects low probability tokens since the spread of importance ratio is higher at lower probabilities. See Appendix A for additional examples of lower clip leading to collapse.

We now analyse training collapse through the local effect of per-token gradient updates and show that negative-advantage low probability tokens reduce entropy. We write the next-token distribution as $p = \text{softmax}(z)$ for logits $z \in \mathbb{R}^V$ over vocabulary $[V]$. Let $A \in \mathbb{R}$ be the sequence-level advantage and consider the advantage-weighted negative log-likelihood, $\ell_s(z) = -A \log p_s$, where $s \in [V]$ is the sampled token. For any non-sampled token $i \neq s$, a small gradient descent step of size $\eta > 0$ yields

$$\Delta p_i = \eta A p_i \left(C(p) - p_s - p_i\right) + O(\eta^2), \quad (1)$$

where $C(p) = \sum_{j=1}^{V} p_j^2$ (proof in Appendix B). We note that in most practical settings the learning rate η is a small number, so this first-order approximation is applicable. Equation 1 shows that a gradient update can increase the probability of non-sampled tokens. For negative-advantage samples, this is expected: suppressing the sampled token redistributes probability mass elsewhere. More surprisingly, it can also occur for positive-advantage samples. When $A > 0$, a non-sampled token increases in probability if $p_s + p_i < C(p)$ and decreases if $p_s + p_i > C(p)$; for $A < 0$, the inequalities reverse. Thus, reinforcing a low-probability successful token need not sharpen the distribution around that continuation, but can instead raise other low-probability alternatives and broaden the local distribution.

The quantity $C(p)$ therefore acts as an adaptive reference level. Motivated by this, we define peak and valley sampled tokens as follows:

$$\text{Peak} : p_s \geq C(p), \quad \text{Valley} : p_s < C(p). \quad (2)$$

Splitting these two cases by the sign of the advantage gives four regimes: Pos-peak, Pos-valley, Neg-peak and Neg-valley.

This token taxonomy is more specific than entropy alone. Entropy is a property of the full next-token distribution and does not depend on which token was sampled. By contrast, the peak/valley distinction is conditioned on the token that is actually sampled. It therefore connects the sampled rollout to the local geometry of the softmax update. Figure 3a,b illustrates the peak/valley categorization under two different probability profiles. Unlike a fixed probability cutoff, $C(p)$ adapts to distribution sharpness: in flatter distributions, many tokens can be peak-like, while in concentrated distributions, only dominant tokens exceed $C(p)$.

The threshold $C(p)$ also satisfies the expected order properties. Let $p_{\text{max}}, p_{\text{min}}$ be the tokens with maximum and minimum probability under p, then

- $p_{\text{max}} \geq C(p)$. Hence every token attaining p_{max} is a peak.
- $p_{\text{min}} \leq C(p)$, with equality only when p is uniform on its support. Hence nearly every minimum-probability token is a valley.

Thus the taxonomy agrees with the intended intuition: high-probability sampled tokens are peak-like, while low-probability sampled tokens are valley-like. The advantage sign then determines whether the update reinforces or suppresses that region of the distribution.

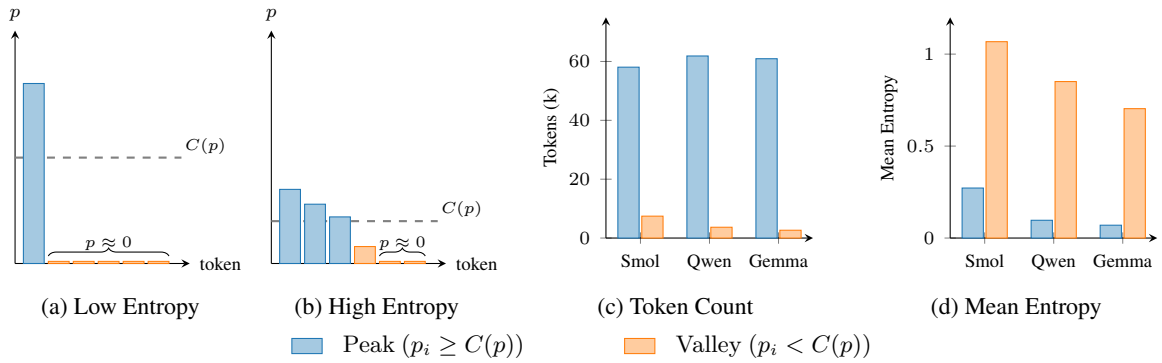


Figure 3: **(a, b):** Equation (2) captures peaks and valleys in both low- and high-entropy distributions. **(c, d)** We report the number of sampled tokens and their average entropy for SmolLM3-3B, Qwen3-4B, and Gemma3-4B on NuminaMath-LEAN. Across models, valley tokens are sampled much less frequently than peak tokens, but have substantially higher entropy.

A similar first-order view also explains the entropy trends induced by the four token types. Let $H(p) = -\sum_i p_i \log p_i$. The change in entropy can be approximated as

$$\Delta H = -\eta A\Big[\Big(p_s \log p_s - \sum_i p_i^2 \log p_i\Big) + H(p)\Big(p_s - C(p)\Big)\Big] + O(\eta^2). \quad (3)$$

For valley tokens, the bracketed factor is negative (Appendix B.3), so Pos-valley updates increase entropy and Neg-valley updates decrease it. The first term inside the bracket is zero in expectation, while for peak tokens the second term is positive per definition in Equation (2). Thus, Pos-peak updates usually decrease entropy, while Neg-peak updates tend to increase it. Appendix B.3 gives the full theoretical discussion; the next section verifies these predictions empirically during RL fine-tuning.

4 Diagnosing Token-Level Instability in RLVR

Isolating the four token regimes. We train SmolLM3-3B on NuminaMath-LEAN while masking the loss to each peak–valley and advantage-sign regime. Figure 4a,b shows distinct reward–entropy dynamics in each case. Pos-peak is stable but quickly plateaus, reinforcing already likely continuations, reducing entropy but offering limited pressure to explore alternative solution paths. Neg-peak and Pos-valley, the two entropy-increasing regimes, lead to collapse: both move mass toward non-sampled alternatives, causing generations to drift toward increasingly random continuations and rewards to deteriorate.

Neg-valley shows a third pattern. It often improves the reward early, but eventually collapses. A negative update on a low-probability token suppresses an unlikely continuation and redistributes mass to other tokens. This can help when the token is spurious, but in high-entropy contexts where valley tokens are common (see Figure 3d), it can also prematurely concentrate mass. The resulting failure is therefore low-entropy: generations become confident, repetitive, or generic (see Figure 1).

Overall, the quadrant experiments (Figure 4a,b) match the entropy predictions of the taxonomy: Pos-peak is stable but conservative; Neg-peak and Pos-valley are entropy-increasing and prone to random collapse; and Neg-valley can help initially but may induce overconfident collapse. Thus, instability depends not only on how far the update moves from the rollout policy, but on which token-level gradients survive the clipping rule. We show similar trends on Qwen3-4B in Appendix C.

Taxonomy-based masking. Pos-peak is stable but fails to reach optimal rewards. An effective policy optimization must utilize the unstable but exploratory quadrants, i.e. Neg-peak or Pos-valley. Group dynamics with more quadrants depends on the prior distribution of a model for a given dataset. However we observe consistent trends along four axis-aligned choices: Pos, Neg, Peak, and Valley. Each combines one entropy-increasing and one entropy-decreasing regime, retaining both exploration and exploitation pressure. Figure 4c shows the resulting training dynamics.

Peak training collapses rapidly despite having the stable Pos-peak component, suggesting that Neg-peak dominates by suppressing high-probability sampled tokens and sharply redistributing mass to alternatives. Neg training learns quickly but later collapses, consistent with prior

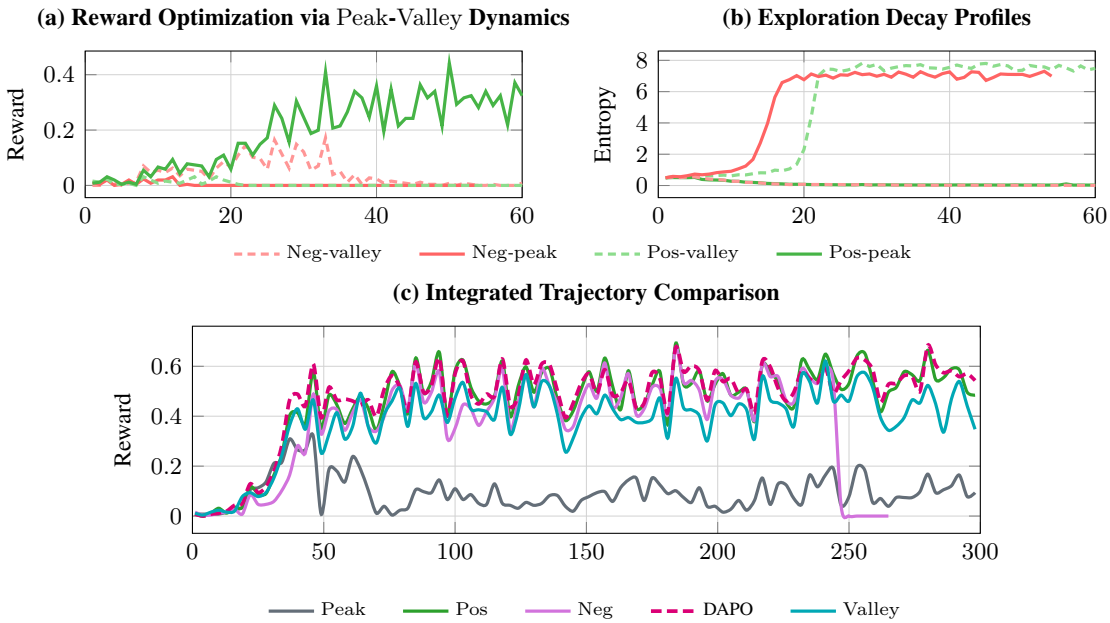


Figure 4: Single-group taxonomy ablations on SmolLM3-3B trained on NuminaMath-LEAN. Panels (a,b) vary peak/valley bounds, showing reward convergence and structural entropy decay. Panel (b) validates Equation (3): Pos-valley and Neg-peak increase entropy, while Pos-peak and Neg-valley decrease it. Entropy-increasing groups collapse rapidly across hyperparameters. Panel (c) plots running-average reward for five primary configurations.

observations (Zhu et al., 2025). Since Neg-peak appears in both masks, it is a plausible source of severe instability. Valley training is more stable and resembles a high-entropy filtering objective (Wang et al., 2025b). This matches the higher-entropy contexts in which valley tokens occur (Figure 3d). However, Valley underperforms both Pos and lenient-clipping baselines, likely because such tokens are sampled less often (see Figure 3c). Valley tokens are also sensitive to ratio-based clipping and are aggressively removed (see Figure 2c).

Overall the advantage sign is the most stable coarse filter. Pos training performs on par with the DAPO baseline, suggesting that rewarded rollouts already contain sufficient learning signal. Unlike clipping or divergence-based constraints, this intervention introduces no additional regularizer or trust-region threshold; it simply restricts which advantage signs are allowed to update the model.

5 Winner Advantage Policy Optimization

From the previous section, we see that positive advantage updates contain both exploitation and rewarded exploration: Pos-peak reinforces high-probability successful tokens, while Pos-valley reinforces successful low-probability tokens. This motivates a simple intervention: remove non-positive advantage terms from the policy-gradient update. We call the resulting method *Winner Advantage Policy Optimization* (WAPO). Unlike rejection fine-tuning, WAPO remains an online GRPO-style policy-gradient method: it uses sampled rollouts, group mean-centered advantages, token-level importance ratios, and clipping.

Binary-reward intuition. We first motivate WAPO in an idealized binary-reward setting. For a fixed prompt x, let $q_x = \Pr_{y\sim\pi_\theta(\cdot|x)}[r(x,y)=1]$ denote the probability that the current policy produces a correct response. Since $r(x,y) \in \{0,1\}$, the expected-reward objective is exactly the success probability:

$$J_x(\theta) = \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r(x,y)] = q_x. \tag{4}$$

Therefore the policy gradient of the binary-reward objective is ∇q_x.

Let $y_i = (y_{i1}, \ldots, y_{iT_i})$ be the i-th rollout with sequence length T_i, $p_{ij} = \pi_\theta(y_{ij} \mid x, y_{i,<j})$ and r_i denote $r(x, y_i)$. With G sampled rollouts, we have (see Appendix D)

$$\nabla q_x \approx \frac{1}{G}\sum_{i=1}^{G}\sum_{j=1}^{T_i} r_i \nabla \log p_{ij}. \tag{5}$$

We now consider a pos-only, group-centered advantage update. Let $\bar{r} = \frac{1}{G}\sum_i r_i$. For successful rollouts, we can define the advantage as:

$$A_i^+ = (r_i - \bar{r})\mathbb{1}[r_i = 1] = (1-\bar{r})r_i. \tag{6}$$

Thus, normalizing by the group size gives

$$\frac{1}{G}\sum_{i=1}^{G}\sum_{j=1}^{T_i} A_i^+ \nabla \log p_{ij} \approx (1-q_x)\nabla q_x, \quad (7)$$

where we use $\bar{r} \approx q_x$ (see Appendix D).

This shows that the pos-only group-normalized update ascends the same direction as the binary policy gradient, but with an adaptive factor $1 - q_x$. A different pos-only normalization using the number of successful samples $G\bar{r}$ yields an adaptive factor of $\frac{1-q_x}{q_x}$. See Appendix D for more details. Both these adaptive factors have an intuitive interpretation: they automatically weight prompts according to success under current policy, putting more weight on harder prompts while attenuating easier ones. They differ in how aggressively they emphasize hard prompts. $\frac{1-q_x}{q_x}$ grows rapidly when q_x is small, which may overemphasize rare or noisy successes under coarse rewards. We therefore use the bounded $1 - q_x$ factor as the default in WAPO.

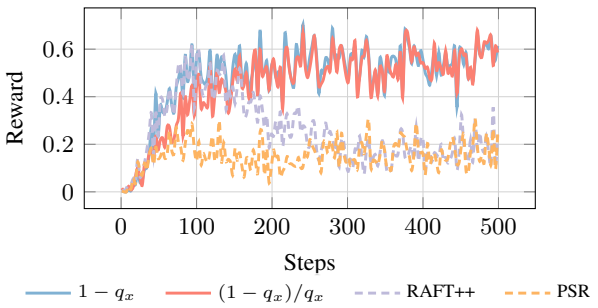
WAPO objective. Generalizing the above analysis to the continuous reward setting, for each prompt x we sample a group of G completions $\{y_i\}_{i=1}^{G}$ from the rollout policy $\pi_{\theta_{\text{old}}}$. Let $A_i^+ = \max(A_i, 0)$, where A_i is the group-normalized advantage of completion i. For token j in rollout i, define the token-level importance ratio $\rho_{ij}(\theta) = \frac{\pi_\theta(y_{ij}|x, y_{i,<j})}{\pi_{\theta_{\text{old}}}(y_{ij}|x, y_{i,<j})}$. We optimize

$$\mathcal{J}_{\text{WAPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\Bigg[\frac{1}{GT}\sum_{i=1}^{G}\sum_{j=1}^{T_i} A_i^+ \cdot \min(\rho_{ij}(\theta), 1+\epsilon) \;\Bigg|\; \exists\, i \text{ s.t. } A_i^+ > 0\Bigg]. \quad (8)$$

Note that the principled policy gradient in Equation 5 does not normalize by the individual sequence length, T_i. We adapt this in Equation 8, where we apply a uniform normalization using a configured maximum sequence length T. This approach is also supported by Liu et al. (2025), who show that this mitigates inherent biases related to sequence length. Since WAPO keeps only $A_i > 0$ terms, the lower clipping bound in Equation (8) is inactive for the retained terms.

Relation to positive-only baselines. WAPO is related to positive-only methods such as PSR (Zhu et al., 2025) and RAFT++ (Xiong et al., 2025), which also remove failed or non-winning completions. However, these methods are mainly formulated for binary winner-selection settings:

(a) Positive-only ablations on NuminaMath-LEAN



(b) Sequence-level normalization short-answer bias. RAFT++ and PSR normalize at the sequence level, giving each selected completion comparable mass regardless of its token length. In late RAFT++ rollouts, generations often collapse to terse answer templates:
`<think> Okay, the answer is 3. So Answer: 3. </think> Answer: 3`

Figure 5: Comparison of WAPO with RAFT++ and PSR on NuminaMath-LEAN using SmolLM3-3B. RAFT++ exhibits short-answer bias, while PSR plateaus without importance ratios.

PSR does not use a clipped group-relative policy-gradient objective, and RAFT++ uses per sequence normalization and does not use mean-centered group-relative advantages. This can give short successful completions disproportionate weight and bias training toward terse answer templates (Figure 5b). In contrast, WAPO keeps the GRPO-style advantage formulation and masks only by the sign of the advantage. Thus, in binary-reward settings it acts as a winner-only update, while in continuous-reward settings it naturally selects completions that score above the group baseline. Figure 5a compares PSR, RAFT++, and the two WAPO normalizations on NuminaMath-LEAN with SmolLM3-3B using exact match as the binary reward. Both WAPO variants outperform positive-only baselines.

6 Results

We conduct extensive experiments across mathematical reasoning and multi-hop QA, providing a well-rounded evaluation across different task formats and model families. Specifically, we evaluate on Math-500 and NuminaMath-LEAN for mathematical reasoning, and Hotpot-QA and OTT-QA for multi-hop QA. In the single-turn math setting, the model directly produces a final answer, while in the multi-turn QA setting, the model interacts with a search environment before answering. We test three comparable-scale model families, Qwen3-4B, SmolLM3-3B, and Gemma3-4B, and compare WAPO against popular RLVR baselines, including GRPO, DAPO, and GSPO where each

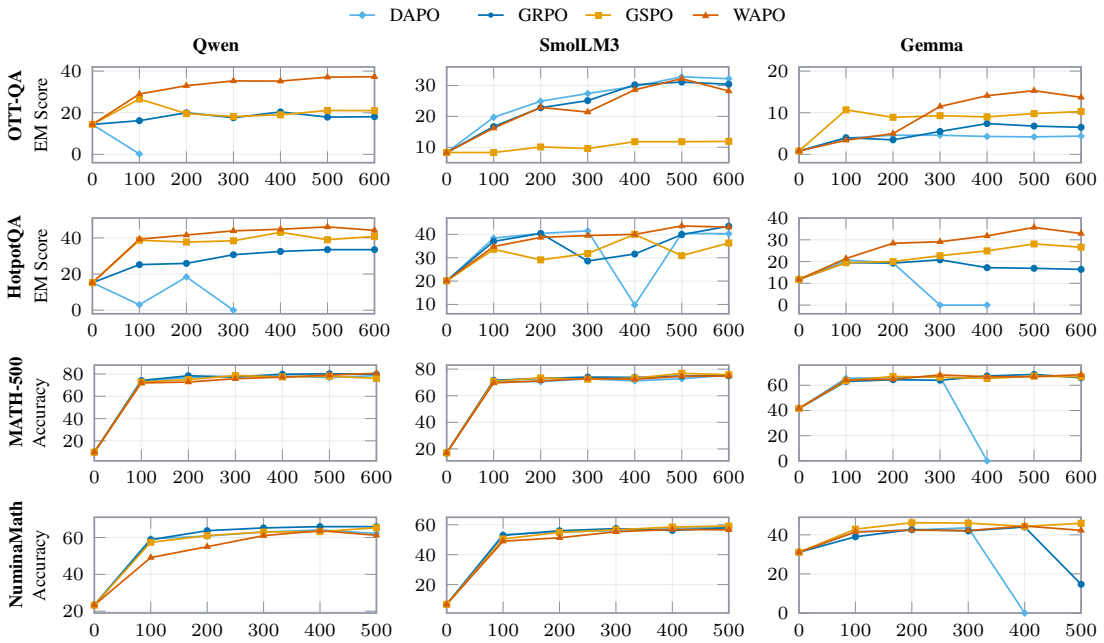


Figure 6: **Evaluation curves for exact match and accuracy.** Rows show datasets, columns show model families. Each subplot reports checkpoint performance up to 600 training steps, with step 0 initialized from the shared base model for math tasks or the SFT-coldstart model for multihop QA. Compared with DAPO, GRPO, and GSPO, WAPO is generally more stable and achieves stronger final performance across settings.

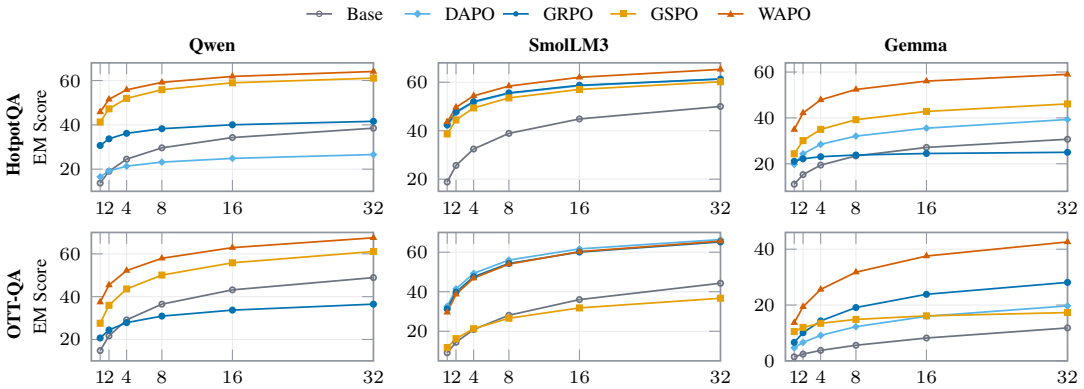


Figure 7: **HotpotQA and OTT-QA pass@k curves.** Each sub-figure plots pass@k EM score as a function of k for one model family on the best evaluation checkpoint. WAPO not only outperforms baselines in pass@1, but also shows good pass@k results. NuminaMath-LEAN pass@k results are shown in appendix.

baseline is tuned to a strong-performing configuration under our experimental setup. For GRPO we find $\epsilon = 9.0$ works best, while for DAPO we set $\epsilon_{\text{low}} = 0.9$, $\epsilon_{\text{high}} = 9.0$, and drop advantage normalization similar to Liu et al. (2025). In the case of GSPO, since it is sequence normalized it is more robust to outliers and we see the best results with $\epsilon = 0.2$. Additional results with parameter tuning are provided in Appendix F. We evaluate performance using exact-match (EM) accuracy, training stability, pass@k for measuring preservation of exploratory behavior, and out-of-domain generalization. Dataset preparation and training details are provided in Appendices G.1, G.2, and G.3.

WAPO improves stability across tasks and model families, with large gains on multi-hop QA and competitive math performance. Figure 6 reports checkpoint performance across four datasets and three model families. WAPO remains stable across all evaluated settings, whereas DAPO exhibits collapse in several cases, and GRPO collapses with Gemma3-4B on the NuminaMath-LEAN dataset. GRPO and GSPO are generally more stable than DAPO, but often saturate early, especially on the multi-hop QA tasks. In contrast, WAPO remains stable and continues

(a) Hotpot-QA→ 2-wiki				(b) Numina → AIME'25			
Method	Qwen	Smol	Gemma	Method	Qwen	Smol	Gemma
DAPO	13.3	30.3	22.0	DAPO	39.06	**27.18**	13.75
GRPO	26.6	28.7	23.3	GRPO	**45.10**	25.93	13.54
GSPO	29.7	29.8	24.4	GSPO	42.08	26.04	**14.27**
WAPO	**34.2**	**31.4**	**25.6**	WAPO	38.43	26.77	12.71

Table 1: **Out-of-domain evaluation.** Best-checkpoint transfer from source to target datasets. We report exact match for 2-wiki and average@32 for AIME'25.

to improve through training, suggesting that removing non-positive advantage updates reduces a major source of RLVR instability without hurting learning capability.

On OTT-QA and Hotpot-QA, WAPO consistently achieves strong performance across model families: on the OTT-QA dataset, WAPO outperforms the next best stable baseline by 9.9% for Qwen3-4B and 3.2% for Gemma3-4B; similarly on the Hotpot-QA dataset, the margins are 4.5% with Qwen3-4B and 10.6% with Gemma3-4B respectively. On Math-500 and NuminaMath-LEAN, WAPO nearly matches other baselines, while slightly lagging on early checkpoints. One possible explanation is the adaptive factor $1 - q_x$ used by WAPO, which emphasizes lower-success prompts and can yield more conservative updates early in training. This pattern is visible on NuminaMath-LEAN in Figure 6: both SmolLM3-3B and Gemma3-4B initially trail the baselines, but close the gap at later training steps. Overall, WAPO appears to be most beneficial on more difficult datasets, as indicated by lower EM score ranges across methods. For example WAPO performs best on OTT-QA (hardest) and remains largely comparable to the strongest baselines on Math-500 (easiest). Table 8 in Appendix G.4 includes exact numbers for the plots in Figure 6.

WAPO preserves exploration. A potential concern is that removing negative-advantage updates may over-exploit already discovered trajectories and reduce sampling diversity. We report Hotpot-QA and OTT-QA pass@k curves at the best evaluated checkpoint in Figure 7. WAPO achieves the best pass@1 and also maintains the strongest pass@k performance across all three model families. We also observe competitive performance at higher pass@k values on both Math-500 and NuminaMath-LEAN. For example, on NuminaMath-LEAN with SmolLM3-3B, WAPO achieves 81.02% pass@16, matching GSPO's 81.0%. (see Appendix G.4, Figure 10).

Model	Method	p@1	p@2	p@4	p@8	p@16	p@32
Qwen3-4B	GRPO	**45.1**	**54.9**	**61.8**	65.9	68.4	70.0
Qwen3-4B	DAPO	39.1	47.5	55.2	62.0	69.7	**80.0**
Qwen3-4B	GSPO	42.1	51.9	60.6	**66.7**	71.3	76.7
Qwen3-4B	WAPO	36.4	45.1	54.3	63.8	**71.5**	76.7
SmolLM3-3B	GRPO	25.9	31.5	38.2	45.8	53.0	60.0
SmolLM3-3B	DAPO	**27.2**	33.4	40.5	47.9	53.9	56.7
SmolLM3-3B	GSPO	26.0	32.6	40.9	50.2	59.1	**66.7**
SmolLM3-3B	WAPO	25.9	**33.5**	**42.9**	**53.5**	**61.7**	**66.7**
Gemma3-4B	GRPO	13.5	16.6	20.3	24.4	**28.9**	**33.3**
Gemma3-4B	DAPO	13.8	17.1	20.5	23.5	26.3	30.0
Gemma3-4B	GSPO	**14.3**	**17.6**	**21.0**	**24.3**	27.4	30.0
Gemma3-4B	WAPO	12.4	15.9	19.6	23.4	25.5	30.0

Table 2: Out-of-domain transfer from NuminaMath-LEAN to AIME'25 pass@k EM-results.

WAPO shows robust out-of-domain transfer. Table 1 evaluates out-of-domain transfer from the training distribution to a held-out target dataset. On Hotpot-QA→2-wiki, WAPO achieves the best performance across all three model families, indicating that its gains on multi-hop QA are not specific to the training benchmark. On NuminaMath-LEAN→AIME'25, WAPO remains competitive with RLVR baselines. The pass@k results in Table 2 show a similar trend to the in-domain setting: although WAPO is not always strongest at low k, it becomes increasingly competitive as k grows, especially for Qwen3-4B and SmolLM3-3B.

7 Conclusion

We presented a token-level gradient analysis of RLVR instability and introduced a peak–valley taxonomy for understanding how advantage-weighted updates shape stability and exploration. Building on this analysis, we proposed WAPO, a minimal GRPO-style objective that removes non-positive advantage updates while retaining online rollouts, group-centered advantages, importance ratios, and clipping. Across mathematical reasoning and multi-hop QA benchmarks, WAPO improves training stability, preserves pass@k, and outperforms strong RLVR baselines in regimes where existing methods collapse or saturate early, while remaining competitive in settings where those baselines are stable. Future work includes extending this study to additional verifiable domains, including programming tasks such as text-to-SQL (Pourreza and Rafiei, 2024; Gorti et al., 2025) and code generation (Le et al., 2022; Chen et al., 2022), as well as to larger model scales and MoE architectures.

Limitations

The empirical evaluation of WAPO is limited in two respects. First, we do not evaluate mixture-of-experts (MoE) models, leaving its behavior under sparse architectures with expert routing and heterogeneous parameter updates unclear. Second, our experiments focus on relatively small model scales and do not include larger frontier-scale models. The scaling behavior of WAPO with respect to model size therefore remains an important direction for future work. Another limitation concerns reward quality. Since WAPO removes non-positive advantage updates, it relies heavily on the correctness of positive-reward or above-baseline completions. In settings where the verifier is noisy, incomplete, or vulnerable to shortcut solutions, WAPO may reinforce false positives more strongly than objectives that also use negative-advantage information. Studying the robustness of positive-only policy-gradient updates under noisy or misspecified rewards is therefore an important direction for future work. We do not foresee any particular societal risks of applying our methods.

References

Elie Bakouch, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Lewis Tunstall, Carlos Miguel Patiño, Edward Beeching, Aymeric Roucher, Aksel Joonas Reedi, Quentin Gallouédec, Kashif Rasul, Nathan Habib, Clémentine Fourrier, Hynek Kydlicek, Guilherme Penedo, Hugo Larcher, Mathieu Morlon, Vaibhav Srivastav, Joshua Lochner, and 4 others. 2025. SmolLM3: smol, multilingual, long-context reasoner. https://huggingface.co/blog/smollm3.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, Chengjun Xiao, Chengyu Du, Chi Zhang, Chu Qiao, Chunhao Zhang, Chunhui Du, Congchao Guo, Da Chen, Deming Ding, and 107 others. 2025. MiniMax-M1: Scaling Test-Time Compute Efficiently with Lightning Attention. *arXiv:2506.13585*.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. 2022. Codet: Code generation with generated tests. In *The Eleventh International Conference on Learning Representations*.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Yang Wang, and William W. Cohen. 2021. Open question answering over tables and text. In *International Conference on Learning Representations*.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. 2026. Beyond Benchmarks: MathArena as an Evaluation Platform for Mathematics with LLMs. *arXiv:2605.00674*.

Wenlong Deng, Yushu Li, Boying Gong, Yi Ren, Christos Thrampoulidis, and Xiaoxiao Li. 2025. On Group Relative Policy Optimization Collapse in Agent Search: The Lazy Likelihood-Displacement. *arXiv:2512.04220*.

Satya Krishna Gorti, Ilan Gofman, Zhaoyan Liu, Jiapeng Wu, Noël Vouitsis, Guangwei Yu, Jesse C Cresswell, and Rasa Hosseinzadeh. 2025. Msc-sql: Multi-sample critiquing small language models for text-to-sql translation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2145–2160.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025. Skywork Open Reasoner 1 Technical Report. *arXiv:2505.22312*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, and 196 others. 2025. Gemma 3 Technical Report. *arXiv:2503.19786*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. volume 35, pages 21314–21328.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding R1-Zero-Like Training: A Critical Perspective. In *Second Conference on Language Modeling*.

Wenhan Ma, Hailin Zhang, Liang Zhao, Yifan Song, Yudong Wang, Zhifang Sui, and Fuli Luo. 2025. Stabilizing MoE Reinforcement Learning by Aligning Training and Inference Routers. *arXiv:2510.11370*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35.

Mohammadreza Pourreza and Davood Rafiei. 2024. Dts-sql: Decomposed text-to-sql with small large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8212–8220.

Project-Numina. 2025. Numinamath-lean: A large-scale lean 4 formalization dataset for competition mathematics. Accessed 2026-05-23.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. 2025. Generalizing verifiable instruction following. In *Advances in Neural Information Processing Systems*, volume 38.

Penghui Qi, Zichen Liu, Xiangxin Zhou, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Defeating the Training-Inference Mismatch via FP16. *arXiv:2510.26788*.

Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. 2026. Rethinking the Trust Region in LLM Reinforcement Learning. *arXiv:2602.04879*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741.

Yi Ren and Danica J. Sutherland. 2025. Learning Dynamics of LLM Finetuning. In *The Thirteenth International Conference on Learning Representations*.

John Schulman and Thinking Machines Lab. 2025. LoRA Without Regret. *Thinking Machines Lab: Connectionism*. Https://thinkingmachines.ai/blog/lora/.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv:2402.03300*.

Haiming Wang, Mert Unsal, Xiaohan Lin, Mantas Baksys, Junqi Liu, Marco Dos Santos, Flood Sung, Marina Vinyes, Zhenzhe Ying, Zekai Zhu, Jianqiao Lu, Hugues de Saxcé, Bolton Bailey, Chendong Song, Chenjun Xiao, Dehao Zhang, Ebony Zhang, Frederick Pu, Han Zhu, and 21 others. 2025a. Kimina-Prover Preview: Towards Large Formal Reasoning Models with Reinforcement Learning. *arXiv:2504.11354*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv:2212.03533*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025b. Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective Reinforcement Learning for LLM Reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pages 115452–115486.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. 2025. A Minimalist Approach to LLM Reasoning: from Rejection Sampling to Reinforce. *arXiv:2504.11343*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv:2505.09388*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Chenlu Ye, Xuanchang Zhang, Yifan Hao, Zhou Yu, Ziji Zhang, Abhinav Gullapalli, Hao Chen, Jing Huang, and Tong Zhang. 2026. Adaptive Layerwise Perturbation: Unifying Off-Policy Corrections for LLM RL. *arXiv:2603.19470*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, juncai liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. In *Advances in Neural Information Processing Systems*, volume 38, pages 113222–113244.

Yaxiang Zhang, Yingru Li, Jiacai Liu, Jiawei Xu, Ziniu Li, Qian Liu, and Haoyuan Li. 2026. Beyond Precision: Training-Inference Mismatch is an Optimization Problem and Simple LR Scheduling Fixes It. *arXiv:2602.01826*.

Chujie Zheng, Kai Dang, Bowen Yu, Mingze Li, Huiqiang Jiang, Junrong Lin, Yuqiong Liu, Hao Lin, Chencan Wu, Feng Hu, An Yang, Jingren Zhou, and Junyang Lin. 2025a. Stabilizing Reinforcement Learning with LLMs: Formulation and Practices. *arXiv:2512.01374*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025b. Group Sequence Policy Optimization. *arXiv:2507.18071*.

Xinyu Zhu, Mengzhou Xia, Zhepei Wei, Wei-Lin Chen, Danqi Chen, and Yu Meng. 2025. The Surprising Effectiveness of Negative Reinforcement in LLM Reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pages 126546–126573.

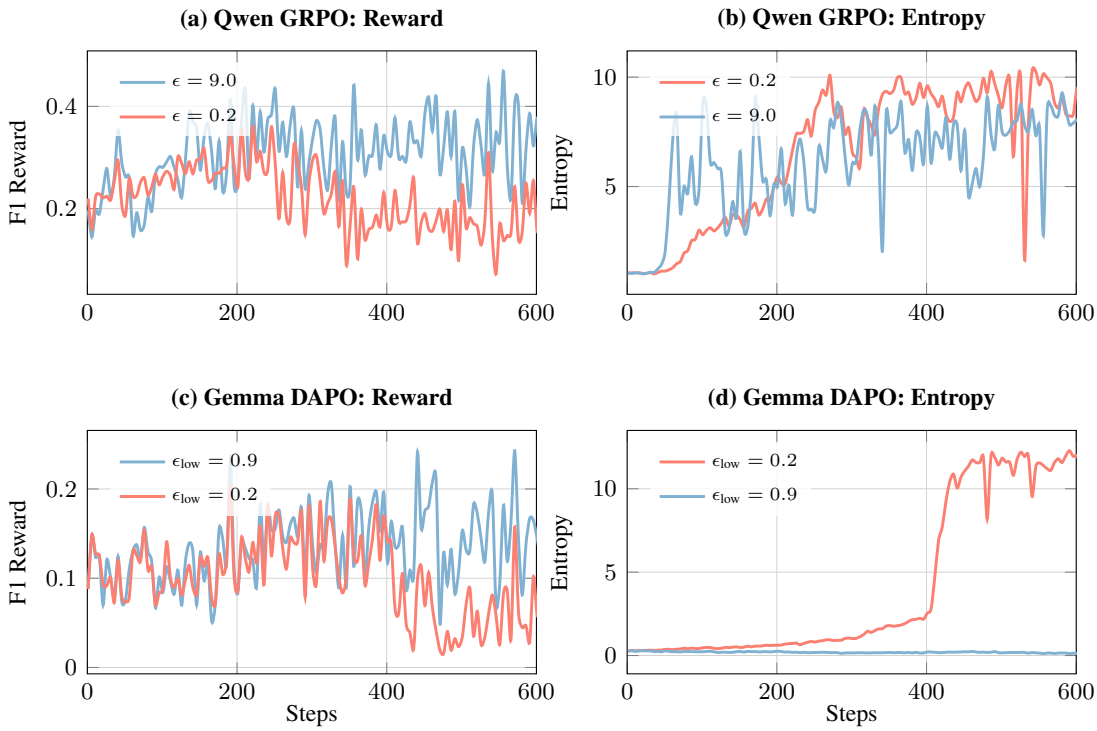


Figure 8: F1-reward and entropy trajectories on OTT-QA under varying negative clipping thresholds. **Top (a, b):** Qwen3-4B trained with GRPO. **Bottom (c, d):** Gemma3-4B trained with DAPO. Lower ϵ_{low} indicates more aggressive clipping (red), which surpresses divergent negative tokens, but does not always improve training stability while increasing entropy.

A Additional Collapse Examples

We show additional examples of aggressive clipping causing collapse in Figure 8. As compared to Figure 2, here we use Qwen3-4B (Yang et al., 2025) trained with GRPO (Deng et al., 2025), and Gemma3-4B (Kamath et al., 2025) trained with DAPO (Yu et al., 2025). In both settings, more aggressive clipping suppresses a larger fraction of divergent tokens but does not improve stability. This highlights that RLVR instability is not explained solely by the trust region.

B Proofs for the Gradient-Motivated Taxonomy

B.1 First-Order Gradient Prediction

Here we derive the result in Equation (1).

Fact: Denote logits as z, the softmax probability of the sampled token as p_s, and similarly p_i for a non-sampled token. A small gradient-descent step with step size $\eta > 0$ on the advantage-weighted negative log-likelihood $\ell_s(z) = -A \log p_s$ changes the probability of p_i as

$$\Delta p_i = \eta A p_i \left(C(p) - p_s - p_i\right) + O(\eta^2), \qquad (9)$$

where $C(p) = \sum_{j=1}^{V} p_j^2$.

Proof. Let z denote the initial logits and z' denote the updated logits after a single gradient descent step. The gradient of $\ell_s(z)$ with respect to an arbitrary logit z_k is computed as:

$$\frac{\partial \ell_s}{\partial z_k} = -A \frac{1}{p_s} \frac{\partial p_s}{\partial z_k} = A(p_k - \delta_{sk}), \qquad (10)$$

where δ_{sk} is the Kronecker delta ($\delta_{sk} = 1$ if $s = k$, and 0 otherwise). Applying the gradient descent update with step size η, the change in the logits is:

$$\Delta z_k = z'_k - z_k = -\eta A(p_k - \delta_{sk}). \qquad (11)$$

To find the resulting change in the probability distribution Δp_i for a non-sampled token $i \neq s$, we perform a first-order Taylor expansion of $p_i(z)$ around z:

$$\Delta p_i = \sum_{k \in [V]} \frac{\partial p_i}{\partial z_k} \Delta z_k + O(\eta^2). \qquad (12)$$

Recall that the Jacobian of the softmax function is given by $\frac{\partial p_i}{\partial z_k} = p_i(\delta_{ik} - p_k)$. Substituting this and

our logit update Δz_k into the expansion yields:

$$\begin{aligned}\Delta p_i &= \sum_{k\in[V]} p_i(\delta_{ik} - p_k)\big(-\eta A(p_k - \delta_{sk})\big) \\ &\quad + O(\eta^2) \\ &= -\eta A p_i \sum_{k\in[V]} (\delta_{ik} - p_k)(p_k - \delta_{sk}) \\ &\quad + O(\eta^2).\end{aligned} \tag{13}$$

We can distribute the terms inside the summation:

$$\begin{aligned}&\sum_{k\in[V]} (\delta_{ik}p_k - \delta_{ik}\delta_{sk} - p_k^2 + p_k\delta_{sk}) \\ &\quad = \sum_k \delta_{ik}p_k - \sum_k \delta_{ik}\delta_{sk} \\ &\quad\quad - \sum_k p_k^2 + \sum_k p_k\delta_{sk}.\end{aligned} \tag{14}$$

Since we are evaluating this for a non-sampled token ($i \neq s$), the term $\delta_{ik}\delta_{sk}$ is uniformly zero for all k. We define the sum of squared probabilities as $C(p) = \sum_{k\in[V]} p_k^2$. Evaluating the remaining sums, we obtain:

$$p_i - 0 - C(p) + p_s. \tag{15}$$

Substituting this simplified summation back into our equation for Δp_i provides the final result:

$$\begin{aligned}\Delta p_i &= -\eta A p_i\big(p_i - C(p) + p_s\big) + O(\eta^2) \\ &= \eta A p_i\big(C(p) - p_s - p_i\big) + O(\eta^2),\end{aligned} \tag{16}$$

which completes the proof. □

B.2 Extremal Tokens and the Reference Level

Next we justify the order properties described in Section 3.

Fact: The reference level $C(p) = \sum_{k\in[V]} p_k^2$ satisfies the following order properties: (1) $p_{\max} \geq C(p)$ and (2) $p_{\min} \leq C(p)$ with equality only for a uniform distribution.

Proof. Because p is a valid probability distribution, we have $\sum_{k\in[V]} p_k = 1$ and $p_k \geq 0$ for all k. We prove each of the three properties in turn.

Property 1: $p_{\max} \geq C(p)$**.**
By definition, $p_k \leq p_{\max}$ for all $k \in [V]$. Multiplying both sides by the non-negative probability p_k yields $p_k^2 \leq p_{\max}p_k$. Summing this inequality over the entire vocabulary gives:

$$\begin{aligned}C(p) &= \sum_{k\in[V]} p_k^2 \leq \sum_{k\in[V]} p_{\max}p_k \\ &= p_{\max} \sum_{k\in[V]} p_k = p_{\max}.\end{aligned} \tag{17}$$

Thus, $p_{\max} \geq C(p)$. It follows immediately that any token attaining the maximum probability satisfies this condition, making it a peak.

Property 2: $p_{\min} \leq C(p)$**.**
Applying the same logic as the Property 1, we note that $p_k \geq p_{\min}$ for all k in the support of p. Multiplying by p_k and summing over k yields:

$$\begin{aligned}C(p) &= \sum_{k\in[V]} p_k^2 \geq \sum_{k\in[V]} p_{\min}p_k \\ &= p_{\min} \sum_{k\in[V]} p_k = p_{\min}.\end{aligned} \tag{18}$$

This establishes that $p_{\min} \leq C(p)$. For equality to hold, $C(p) - p_{\min} = 0$, which implies:

$$\sum_{k\in[V]} p_k(p_k - p_{\min}) = 0. \tag{19}$$

Since $p_k \geq 0$ and $p_k - p_{\min} \geq 0$ for all k, every term in the summation must independently be zero. This requires $p_k = p_{\min}$ for every token in the support, which only occurs when the distribution p is strictly uniform. Therefore, a minimum-probability token is always a valley, except when the distribution is perfectly uniform. □

B.3 First-Order Entropy Direction

Below we derive the change in entropy from Equation (3).

Proof of ΔH Update. Let $H(p) = -\sum_{k\in[V]} p_k \log p_k$ be the entropy of the next-token distribution. The first-order Taylor expansion of the entropy change is:

$$\Delta H = \sum_{k\in[V]} \frac{\partial H}{\partial p_k}\Delta p_k + O(\eta^2). \tag{20}$$

Taking the derivative of $H(p)$, we have $\frac{\partial H}{\partial p_k} = -1 - \log p_k$. Because probability mass must be conserved ($\sum_k p_k = 1$), the sum of the changes is zero: $\sum_{k\in[V]} \Delta p_k = 0$. Thus, the entropy update simplifies to:

$$\Delta H = -\sum_{k\in[V]} \Delta p_k \log p_k + O(\eta^2). \tag{21}$$

From our earlier derivation, the general update for any token k (including the sampled token s) is $\Delta p_k = \eta A p_k(C(p) - p_s - p_k + \delta_{sk})$. Substituting this into our simplified expression for ΔH yields:

$$\begin{aligned}\Delta H &= -\eta A \sum_{k\in[V]} p_k \log p_k \\ &\quad \times \big(C(p) - p_s - p_k + \delta_{sk}\big) + O(\eta^2).\end{aligned} \tag{22}$$

We distribute the summation over the terms inside the parentheses to get:

$$\Delta H = -\eta A \Bigg[(C(p) - p_s) \sum_{k\in[V]} p_k \log p_k - \sum_{k\in[V]} p_k^2 \log p_k + \sum_{k\in[V]} p_k \log p_k \delta_{sk} \Bigg] + O(\eta^2). \tag{23}$$

Recognizing that $\sum_k p_k \log p_k = -H(p)$ and applying the Kronecker delta to the final term, we arrive at the final expression after rearrangement:

$$\Delta H = -\eta A \Bigg[p_s \log p_s - \sum_{k\in[V]} p_k^2 \log p_k + H(p)\big(p_s - C(p)\big) \Bigg] + O(\eta^2). \tag{24}$$

□

Proof of Entropy Direction. Let $B(p_s)$ denote the bracketed term in Equation 24. In Section 3 we described the behaviour of $B(p_s)$ for the token types defined in Equation (2). Here we provide the complete reasoning by computing its sign for valley and peak tokens. For convenience, we define $M(p) = -\sum_{k\in[V]} p_k^2 \log p_k$ and the continuous function $h(x) = -x \log x$.

Direction for valley **tokens ($p_s < C(p)$).** We extend the bracketed term to a function over $x \in [0, 1]$:

$$B(x) = M(p) - h(x) + H(p)\big(x - C(p)\big). \tag{25}$$

Note that we consider this as a function of variable x with p representing a fixed probability distribution, such that $M(p)$, $H(p)$, and $C(p)$ are independent of x. Then the second derivative is $B''(x) = 1/x > 0$, implying $B(x)$ is strictly convex on $(0, 1]$. By convexity, for any $x \in [0, C(p)]$ (noting that $C(p) \le 1$), $B(x)$ is bounded above by the convex combination of its endpoints. Letting $\lambda = x/C(p)$, we have:

$$B(x) \le (1 - \lambda)B(0) + \lambda B(C(p)). \tag{26}$$

We evaluate both endpoints to show they are non-positive. At $x = 0$, we observe:

$$\begin{aligned} B(0) &= M(p) - H(p)C(p) \\ &= \mathbb{E}_{k\sim p}[-p_k \log p_k] \\ &\quad - \mathbb{E}_{k\sim p}[-\log p_k]\mathbb{E}_{k\sim p}[p_k] \\ &= \mathrm{Cov}_{k\sim p}(p_k, -\log p_k) \le 0, \end{aligned} \tag{27}$$

which holds because $-\log$ is a monotonically decreasing function.

At $x = C(p)$, we define a valid probability distribution q where $q_k = p_k^2/C(p)$. Then:

$$\begin{aligned} B(C(p)) &= M(p) - h(C(p)) \\ &= -\sum_{k\in[V]} p_k^2 \log \frac{p_k}{C(p)} \\ &= -C(p) \sum_{k\in[V]} q_k \log \frac{q_k}{p_k} \\ &= -C(p) D_{\mathrm{KL}}(q\|p) \le 0, \end{aligned} \tag{28}$$

Since the KL divergence is non-negative.

Having established that both endpoints of $B(x)$ are non-positive, by Equation (26) we find that $B(x) \le 0$ for all $x \in [0, C(p)]$. Since a valley token satisfies $p_s < C(p)$, it immediately follows that $B(p_s) \le 0$.

Direction for Maximum-Probability peak **tokens.**
Consider a token s where $p_s = \max_k p_k$. From Property 1 in Equation (17), we know $C(p) \le p_s$. We rewrite the bracket $B(p_s)$ by extracting terms:

$$B(p_s) = \sum_{k\in[V]} p_k\big(p_k + p_s - C(p)\big)(-\log p_k) - p_s(-\log p_s). \tag{29}$$

Notice that the weights $w_k = p_k(p_k + p_s - C(p))$ are strictly non-negative and sum exactly to p_s:

$$\begin{aligned} \sum_{k\in[V]} w_k &= \sum_k p_k^2 + p_s \sum_k p_k - C(p) \sum_k p_k \\ &= C(p) + p_s - C(p) = p_s. \end{aligned} \tag{30}$$

Furthermore, because p_s is the maximum probability, $-\log p_k \ge -\log p_s$ for all k. Applying this inequality directly to our expanded bracket yields:

$$\begin{aligned} B(p_s) &= \sum_{k\in[V]} w_k(-\log p_k) - p_s(-\log p_s) \\ &\ge \sum_{k\in[V]} w_k(-\log p_s) - p_s(-\log p_s) \\ &= p_s(-\log p_s) - p_s(-\log p_s) = 0. \end{aligned} \tag{31}$$

Thus, $B(p_s) \ge 0$ for maximum-probability tokens.

Remark: A peak token that is not the global maximum may still yield a negative $B(p_s)$. For

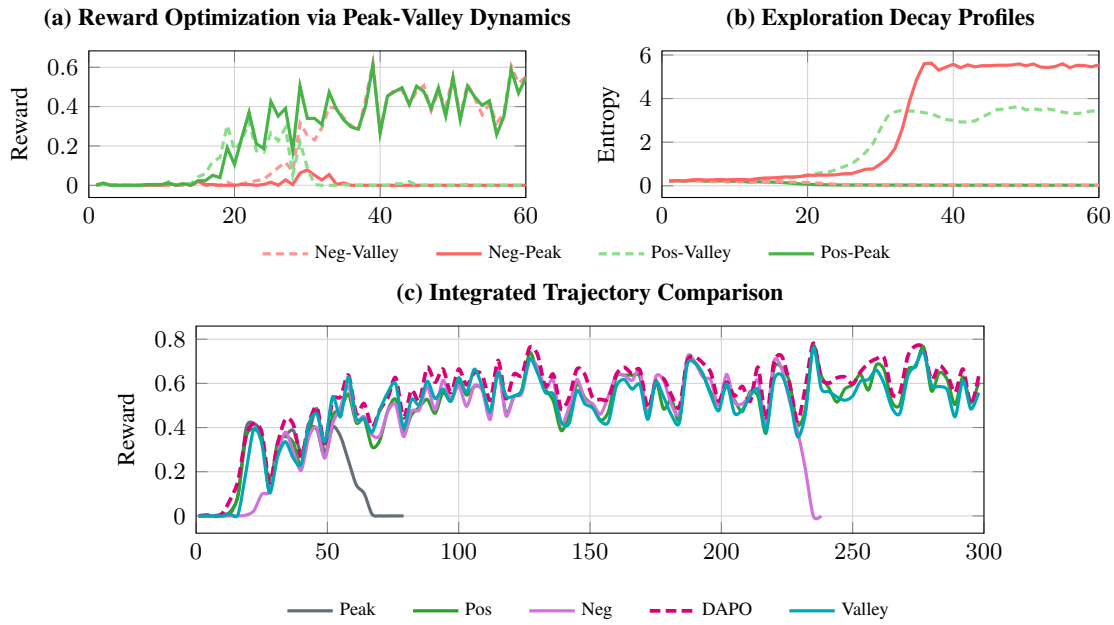


Figure 9: We train Qwen3-4B on NuminaMath-LEAN with one group of our taxonomy at a time. Panel (b) validates Equation (3): Pos-valley and Neg-peak tokens increase entropy, while Pos-peak and Neg-valley tokens decrease entropy. Entropy-increasing groups collapse immediately in (a). Pos-peak is stable but does not match the best integrated trajectory in (c), motivating objectives that balance exploration and exploitation. We observed that the entropy of Qwen3-4B is lower than SmolLM3-3B. This could explain why Neg-valley did not collapse immediately. In contrast, valley-only performs on-par with the baseline DAPO in this setting, showing that it is a promising candidate for future study. Pos-only remains competitive in all settings.

example, if $p = (0.14, 0.46, 0.40)$ and the sampled token is $p_s = 0.40$, then $C(p) \approx 0.391$. While $p_s > C(p)$ (making it a peak), computing the bracket gives $B(p_s) \approx -0.0083 < 0$. However, these corner cases are rare in practice (refer Table 3), and our experiments follow the expected trend that Pos-peak tokens decrease entropy while Neg-peak tokens increase entropy. □

Base model	Peak tokens	$B(p_s) < 0$ tokens	Violation rate
Qwen3-4B	25,976	37	0.14%
SmolLM3-3B	8,870	45	0.50%

Table 3: Peak tokens following the opposite predicted entropy trend at step 0 of the NuminaMath-LEAN experiment. Counts are aggregated across positive- and negative-advantage tokens for each base model.

C Additional Token Dynamics Plots

We include additional entropy and reward plots for NuminaMath-LEAN runs on Qwen3-4B. These plots confirm the entropy direction predictions from Section 3 in Figure 9: pos-valley and neg-peak updates increase entropy, while pos-peak and neg-valley updates decrease entropy.

D WAPO Gradient Analysis

We begin by deriving Equation (5) from Section 5.

Proposition 1. *We aim to prove that for a policy model π_θ operating under a binary reward setting ($r_i \in \{0, 1\}$), the gradient of the probability q_x of generating a correct answer can be empirically estimated over G rollouts as:*

$$\nabla q_x \approx \frac{1}{G} \sum_{i=1}^{G} \sum_{j=1}^{T_i} r_i \nabla \log p_{ij}. \tag{32}$$

Proof. Let $y_i = (y_{i1}, y_{i2}, \ldots, y_{iT_i})$ denote a sequence of T_i tokens generated by the policy model π_θ in response to a prompt x. The joint probability of generating the entire sequence y_i is the product of the individual step-wise token probabilities:

$$P(y_i) = \prod_{j=1}^{T_i} \pi_\theta(y_{ij} \mid y_{i,j-1} \ldots y_{i1}, x) = \prod_{j=1}^{T_i} p_{ij}. \tag{33}$$

Because the reward $r(x, y_i)$ (which we abbreviate as r_i) is binary, taking values in $\{0, 1\}$, the probability q_x that the model correctly answers the prompt is exactly equivalent to the expected reward over

all possible generated sequences:

$$q_x = \mathbb{E}_{y_i \sim \pi_\theta}[r_i] = \sum_{y_i} r_i P(y_i). \tag{34}$$

We compute the gradient of q_x with respect to the model parameters θ. Since the reward function evaluates the correctness of a sequence and does not intrinsically depend on θ, the gradient operator only applies to the sequence probability:

$$\nabla q_x = \sum_{y_i} r_i \nabla P(y_i). \tag{35}$$

We apply the standard log-derivative trick, $\nabla P(y_i) = P(y_i) \nabla \log P(y_i)$, to rewrite this summation back into the form of an expectation:

$$\begin{aligned} \nabla q_x &= \sum_{y_i} r_i P(y_i) \nabla \log P(y_i) \\ &= \mathbb{E}_{y_i \sim \pi_\theta}\left[r_i \nabla \log P(y_i) \right]. \end{aligned} \tag{36}$$

Next, we expand the log probability of the full sequence. Because the sequence probability is a product of stepwise conditional probabilities, its logarithm is a sum:

$$\log P(y_i) = \log \left(\prod_{j=1}^{T_i} p_{ij} \right) = \sum_{j=1}^{T_i} \log p_{ij}. \tag{37}$$

Substituting this expansion into our expected gradient yields the exact analytical policy gradient:

$$\nabla q_x = \mathbb{E}_{y_i \sim \pi_\theta} \left[\sum_{j=1}^{T_i} r_i \nabla \log p_{ij} \right]. \tag{38}$$

In practice, GRPO and similar reinforcement learning algorithms approximate this exact expectation using Monte Carlo sampling. By drawing a group of G independent rollouts $\{y_1, y_2, \ldots, y_G\}$ from the policy π_θ, we replace the expectation with the empirical mean over the G samples, yielding the final objective function gradient used to update the policy:

$$\nabla q_x \approx \frac{1}{G} \sum_{i=1}^{G} \sum_{j=1}^{T_i} r_i \nabla \log p_{ij}. \tag{39}$$

□

Proposition 2. *Consider a positive-only GRPO-style policy gradient method with binary rewards $r_i \in \{0, 1\}$. In a positive-only setting, updates are restricted to successful trajectories, meaning the advantage is effectively masked by the reward: $A_i^+ = (r_i - \bar{r}) r_i$, where $\bar{r} = \frac{1}{G} \sum_{k=1}^{G} r_k$ is the group mean reward. Assuming the empirical mean $\bar{r}$ closely approximates the true success probability q_x (which we abbreviate as q), the gradient objective satisfies:*

$$\frac{1}{D} \sum_{i=1}^{G} \sum_{j=1}^{T_i} A_i^+ \nabla \log p_{ij} \approx \alpha \cdot \nabla q_x, \tag{40}$$

where the evaluated term $\alpha \cdot \nabla q_x$ corresponds to the choices of D as follows:

D	*Evaluated* $\alpha \cdot \nabla q_x$
G	$(1-q)\nabla q_x$
$G \cdot \bar{r}$	$\frac{1-q}{q} \nabla q_x$

Proof. Let $P(y_i) = \prod_{j=1}^{T_i} p_{ij}$ be the probability of generating the sequence y_i. By bringing the inner sum over j inside the derivative and log we find

$$\sum_{j=1}^{T_i} \nabla \log p_{ij} = \nabla \log P(y_i). \tag{41}$$

Substituting this into the left-hand side (LHS) of Equation (40) yields:

$$\text{LHS} = \frac{1}{D} \sum_{i=1}^{G} A_i^+ \nabla \log P(y_i). \tag{42}$$

Because rewards are binary, the advantage only applies when $r_i = 1$, and we can rewrite $A_i^+ = (1 - \bar{r}) r_i$. Substituting this simplified advantage back into our sum,

$$\text{LHS} = \frac{1 - \bar{r}}{D} \sum_{i=1}^{G} r_i \nabla \log P(y_i). \tag{43}$$

From Equation 35 in our previous proof, we established that the empirical estimate for the gradient of the success probability is $\nabla q_x \approx \frac{1}{G} \sum_{i=1}^{G} r_i \nabla \log P(y_i)$. Rearranging this relationship gives:

$$\sum_{i=1}^{G} r_i \nabla \log P(y_i) \approx G \nabla q_x. \tag{44}$$

Substituting this into our LHS equation yields

$$\text{LHS} \approx \frac{G(1 - \bar{r})}{D} \nabla q_x. \tag{45}$$

We now assume that the empirical mean reward over the group G approximates the true probability of generating a correct answer, such that $\bar{r} \approx q$. Applying this approximation, we arrive at the general form:

$$\text{LHS} \approx \frac{G(1-q)}{D} \nabla q_x. \tag{46}$$

Finally, we evaluate this expression for the two specific choices of the normalizer D:

Case 1: $D = G$.
Substituting $D = G$ into the general form, the G terms cancel out:

$$\text{LHS} \approx (1-q) \nabla q_x. \tag{47}$$

This confirms the first row of the table.

Case 2: $D = G \cdot \bar{r}$.
Using our approximation $\bar{r} \approx q$, we substitute $D \approx G \cdot q$:

$$\text{LHS} \approx \frac{1-q}{q} \nabla q_x. \tag{48}$$

This confirms the second row of the table, completing the proof. □

E Baseline Formulations

Here we review the baselines that we compare WAPO against. Let x be a prompt, $\{y_i\}_{i=1}^{G}$ the sampled completions for x, and $y_{i,t}$ the t-th token of completion y_i. We define the token-level importance ratio:

$$\rho_{it}(\theta) = \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x, y_{i,<t})}.$$

For GRPO and GSPO, advantages are group-normalized:

$$A_i = \frac{r_i - \bar{r}}{\text{std}(\{r_j\}_{j=1}^{G}) + \epsilon},$$
$$\bar{r} = \frac{1}{G} \sum_{j=1}^{G} r_j.$$

GRPO The GRPO objective uses token-level clipped importance ratios:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min\Big(\rho_{it}(\theta) A_i, \text{clip}\big(\rho_{it}(\theta), 1-\epsilon, 1+\epsilon\big) A_i \Big).$$

GSPO GSPO replaces token-level ratios with a sequence-level ratio. In our implementation, the sequence ratio is the geometric mean of token ratios:

$$\rho_i^{\text{seq}}(\theta) = \exp\left(\frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \log \rho_{it}(\theta) \right).$$

The objective is:

$$\mathcal{L}_{\text{GSPO}}(\theta) = -\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min\Big(\rho_i^{\text{seq}}(\theta) A_i, \text{clip}\big(\rho_i^{\text{seq}}(\theta), 1-\epsilon, 1+\epsilon\big) A_i \Big).$$

DAPO For DAPO, advantages are mean-centered but not variance-normalized:

$$A_i^{\text{DAPO}} = r_i - \bar{r}.$$

The objective uses token-level ratios with asymmetric clipping:

$$\mathcal{L}_{\text{DAPO}}(\theta) = -\frac{1}{|\mathcal{G}|} \sum_{x \in \mathcal{G}} \frac{1}{\sum_{i=1}^{G} |y_i|} \sum_{i=1}^{G} \sum_{t=1}^{|y_i|} \min\Big(\rho_{it}(\theta) A_i^{\text{DAPO}}, \text{clip}\big(\rho_{it}(\theta), \epsilon_{\text{low}}, \epsilon_{\text{high}}\big) A_i^{\text{DAPO}} \Big).$$

F Additional Results on Clipping Strength

We tuned GRPO with $\epsilon \in \{0.2, 9.0\}$ and DAPO with $(\epsilon_{\text{low}}, \epsilon_{\text{high}}) \in \{(0.2, 9.0), (0.9, 9.0)\}$. We observed that weaker clipping delayed collapse or remained stable as compared to their aggressive counterpart in all but one (OTT-Qwen-DAPO) setting. We mark collapse where reward falls by at least 50% within 50 consecutive steps. The onset of collapse is reported in Table 4. In addition, we observed that in the following settings, weak clipping remained stable, whereas the aggressive clipping baseline collapsed. Across our experiments, we did

Model	Dataset	Loss	$(\epsilon_{\text{low}}, \epsilon_{\text{high}})$	Collapse Onset
Qwen3-4B	OTT-QA	DAPO	$(0.9, 9.0)$	30
Qwen3-4B	OTT-QA	DAPO	$(0.2, 9.0)$	600
Gemma3-4B	HotpotQA	DAPO	$(0.2, 9.0)$	160
Gemma3-4B	HotpotQA	DAPO	$(0.9, 9.0)$	250
Gemma3-4B	Math-500	DAPO	$(0.2, 9.0)$	150
Gemma3-4B	Math-500	DAPO	$(0.9, 9.0)$	320

Table 4: Onset of collapse under different DAPO clipping configurations. Collapse is marked when reward falls by at least 50% within 50 consecutive steps.

not observe any case where aggressive clipping was stable while weak clipping in the same setting collapsed. These settings are summarized in Table 5. Tighter clipping does not consistently improve stability or final performance and, in some settings, slows learning or induces collapse, consistent with Figure 2. GSPO uses sequence-level clipping and was comparatively robust with its standard $\epsilon = 0.2$ configuration. Table 6 reports train reward, computed as a running average over 20 steps, across datasets with SmolLM3-3B.

Loss Function	Dataset	Model
GRPO	HotpotQA	Qwen3-4B, Gemma3-4B
GRPO	OTT-QA	Qwen3-4B, SmolLM3-3B
DAPO	HotpotQA	Qwen3-4B
DAPO	Math	Qwen3-4B
DAPO	OTT-QA	SmolLM3-3B, Gemma3-4B

Table 5: Settings in which weak clipping remained stable while aggressive clipping collapsed.

G Experiment Details

All our experiments are conducted on two NVIDIA A6000 GPUs, where we use one GPU for vllm rollout generation and one GPU for policy updates.

G.1 Math Environment

For mathematical reasoning, each example consists of a problem x and a ground-truth answer a. The model produces a response y, which must contain both a reasoning segment and a final answer segment. We append the following instruction to each math problem:

> Put your reasoning inside `<think>...</think>` tags, then write your final answer as: `Answer: <your answer>`.

The required response format is

$$y = \texttt{<think>}\ \text{reasoning}\ \texttt{</think>}$$
$$\texttt{Answer:}\ \ \text{answer}.$$

The parser extracts the text following `Answer:` as the model's final answer $\hat{a}(y)$. The format indicator is

$$I_{\text{fmt}}(y) = \mathbf{1}\{y \in \mathcal{Y}_{\text{fmt}}\},$$

where $\mathcal{Y}_{\text{fmt}}$ is the set of responses matching the required format. The answer correctness indicator is

$$I_{\text{ans}}(y, a) = \mathbf{1}\{\hat{a}(y) = a\}.$$

The math reward is binary:

$$R_{\text{math}}(y, a) = I_{\text{fmt}}(y) I_{\text{ans}}(y, a).$$

A response receives reward 1 only when it follows the required format and the extracted final answer exactly matches the ground-truth answer; otherwise it receives reward 0.

G.1.1 Math-500 dataset.

For the Math-500 experiments, we use the augmented split released with PRM800K (Lightman et al., 2024). This split was constructed to reduce overfitting to the original 7,500-problem Math-500 training set: 4,500 problems from the original Math-500 test split are added to training, yielding 12,000 training problems, and the remaining 500 test problems are held out for evaluation. Each example contains a problem statement, reference solution, final answer, subject, difficulty level, and unique identifier. We use the problem statement and final answer only. Each problem is converted into a single-turn chat prompt.

G.1.2 NuminaMath-LEAN dataset.

We start from the NuminaMath-LEAN training split (Project-Numina, 2025; Wang et al., 2025a). The raw split contains 104,155 examples. We filter examples before training as follows: keep only examples whose question type is either math-word-problem or; drop examples with missing or empty problem; drop examples with missing or empty answer; and keep only answers that are simple numeric strings, namely integers, decimals, or plain fractions matching forms such as $3, -2, 4.5$, or $7/8$. This removes symbolic answers, interval answers, LaTeX fractions such as $\frac{1}{2}$, multiple-answer strings, and labels such as unknown. After filtering, the dataset contains 21,251 examples: 18,273 math-word-problem examples and 2,978 multiple-choice-question examples. We shuffle the filtered dataset with a fixed seed. By default, the first 1,000 shuffled examples are used as evaluation data and the remaining 20,251 examples are used for training.

G.2 Multi-hop QA Environment

For retrieval-based multi-hop question answering, each example consists of a question x, a gold answer a, and access to a retrieval corpus. The model interacts with the environment over multiple turns. We adopted Search-R1(Jin et al., 2025) style environment interaction, the model will conduct reasoning inside <think>...</think>, issue search query in <search>...</search> and eventually produce a final answer in <answer>...</answer>.

The system prompt used for multi-hop QA has

Dataset	GRPO $\epsilon = 0.2$	GRPO $\epsilon = 9$	DAPO $\epsilon_{\text{low}} = 0.2$ $\epsilon_{\text{high}} = 0.28$	DAPO $\epsilon_{\text{low}} = 0.2$ $\epsilon_{\text{high}} = 9.0$	DAPO $\epsilon_{\text{low}} = 0.9$ $\epsilon_{\text{high}} = 9.0$
ott-qa (f1-reward, step-500)	0.23	0.49	0.36	0.33	0.45
hotpotqa (f1-reward, step-500)	0.34	0.60	0.49	0.41	0.57
numinamath-lean (em, step-500)	0.54	0.55	0.56	0.56	0.58
math-500 (em, step-500)	0.77	0.79	0.73	0.74	0.78

Table 6: Train reward, reported as a running average over 20 steps, across datasets with SmolLM3-3B.

more instructions compared to Search-R1 to prevent hallucinations. We set max_turns to 10 in all multihop QA training and evaluation.:

> You are a helpful assistant. **Your task:**
>
> You need to answer complex questions by retrieving relevant information and reasoning step-by-step.
>
> You **must** conduct reasoning inside `<think>` and `</think>` in each of your responses before you perform a search or give the final answer. The reasoning should include analysing the question, making a plan, and answering sub-questions by reasoning on existing content.
>
> You should obtain knowledge by calling a search engine by `<search> query </search>` when you come up with a suitable query. The responses from the search engine will be the top search results and will be between the tags `<information>` and `</information>`.
>
> Always ground your answer to the information the search engine returns. Do not use any information that is not returned by the search engine. You can call the search engine any number of times you want.
>
> Once enough information is gathered, produce a precise and concise answer to the original question and wrap it in `<answer>...</answer>`.

Search turns use:

$y_t =$ <think> reasoning </think>
<search> query </search>.

Final-answer turns use:

$y_t =$ <think> reasoning </think>
<answer> answer </answer>.

Retrieved evidence is inserted back into the conversation between `<information>` and `</information>` tags.

Let $\hat{a}(y)$ be the extracted final answer. We normalize both prediction and gold answer by lowercasing, removing punctuation and articles, and collapsing whitespace. The F1 reward is computed over normalized answer tokens as

$$R_{\text{F1}} = \frac{2PR}{P + R},$$

where P and R are token precision and recall between $\hat{a}(y)$ and a. If the prediction and gold answer have no shared normalized tokens, $R_{\text{F1}} = 0$.

We also use a format reward $R_{\text{fmt}}(y)$, which checks whether assistant messages follow the required structured search or answer format. The final multi-hop QA reward is

$$R_{\text{QA}}(y, a) = \frac{0.3\, R_{\text{fmt}}(y) + R_{\text{F1}}(y, a)}{1.3}.$$

G.2.1 OTT-QA

We used a subset of 20,000 samples from the OTT-QA (Chen et al., 2021) training dataset to generate high quality trajectories for supervised fine-tuning (SFT) (Ouyang et al., 2022). Among the generated trajectories, we applied rejection sampling to keep trajectories with valid searches, correct formats, and correct answers. We constructed a RL dataset using a disjoint subset of 10,000 samples from the OTT-QA training dataset. To ensure exploration during RL we ran 6 rollouts for each question on a Qwen-SFT model and only kept questions with accuracy between 0% and 50%, exclusive. All OTT-QA checkpoints were evaluated on a held-out test set of 1000 samples.

We created a dense retriever over the OTT-QA table and passage corpus. Each Wikipedia passage is represented by its normalized title plus passage text, while each table is serialized using its title, section title, introduction, and markdown-formatted table content. We did not apply a chunking strategy, and all documents were encoded with the BAAI/bge-m3 embedding model (Chen et al., 2024).

Table 7: Training hyperparameters for math and multi-hop QA experiments. Batch size here means the number of rollouts we generated for one batch, micro-batch size means number of rollouts trained on one GPU.

Math	
Hyperparameter	Value
LoRA rank	8
LoRA alpha	32
Learning rate	1×10^{-5}
Batch size	256
Micro-batch size	1
Mini-batch size	64
Rollouts per prompt	8
Max response length	4096
Max sequence length	8192
Max grad norm	1.0

Multi-hop QA	
Hyperparameter	Value
Max Turns	10
Retriever Top k	3
LoRA rank	8
LoRA alpha	32
Learning rate	1×10^{-5}
Batch size	256
Micro-batch size	1
Mini-batch-size	64
Rollouts per prompt	8
Max response length	1024
Max sequence length	8192
Max grad norm	1.0

G.2.2 Hotpot-QA & 2wiki

Similar to OTT-QA, we used a subset of 10,000 samples from HotpotQ (Yang et al., 2018) to generate a SFT coldstart dataset with the rest of the data used for RL training. Hotpot-QA checkpoints are evaluated on a held-out test set that contains 1000 samples. In our OOD experiments, we evaluated Hotpot-QA checkpoints on a 1000 sample subset of the 2wiki (Ho et al., 2020) test dataset.

We follow the Search-R1 (Jin et al., 2025) instructions to setup a retriever using the wiki-18 (Karpukhin et al., 2020) corpus, and an E5 (Wang et al., 2022) embedding model for HotpotQA and 2wiki training and evaluation.

G.3 Hyperparameters

Table 7 summarizes the training hyperparameters. We use the same settings across different model checkpoints within each task family. We use a larger learning rate 1×10^{-5} because Schulman and Lab (2025) proposed LoRA RL training needs $10\times$ the learning rate comparing to full fine-tuning.

G.4 Supplementary Experiments

Table 8 summarizes the main in-distribution results. On multi-hop QA, WAPO achieves the strongest performance across model families, while DAPO collapses in 4 out of 6 runs. Notably, DAPO collapses within 100 steps on OTT-QA with Qwen3-4B, so we do not report its score. On math benchmarks, WAPO is competitive but trails GRPO or GSPO on some Qwen3-4B and SmolLM3-3B runs, while consistently outperforming DAPO. Additionally we include pass@k curves on the math datasets in Figure 10 where we see competitive performance in most settings. Overall, WAPO offers the strongest stability-performance tradeoff across tasks and model families.

Table 8: **Main in-distribution results.** We report average@32 at the best checkpoint, using EM for OTT-QA/Hotpot-QA and accuracy for Math-500/NuminaMath-LEAN. A red × marks collapse, where reward falls near zero. N/A shows failure within 100 steps.

Dataset	Model	DAPO	GRPO	GSPO	WAPO
OTT-QA	Qwen	N/A ×	20.70	27.57	**37.44**
	Smol	**32.63**	31.27	11.76	29.59
	Gemma	4.67	6.62	10.50	**13.67**
HotpotQA	Qwen	16.56 ×	30.69	41.30	**45.84**
	Smol	42.01 ×	42.44	38.66	**43.62**
	Gemma	19.72 ×	21.09	24.36	**34.91**
MATH-500	Qwen	78.01	**79.51**	78.62	78.90
	Smol	75.66	75.01	**75.88**	74.86
	Gemma	66.50 ×	**67.24**	66.88	66.90
NuminaMath	Qwen	63.02	64.83	**65.17**	61.99
	Smol	**58.79**	58.02	58.40	57.63
	Gemma	42.65 ×	43.39 ×	**44.27**	42.78

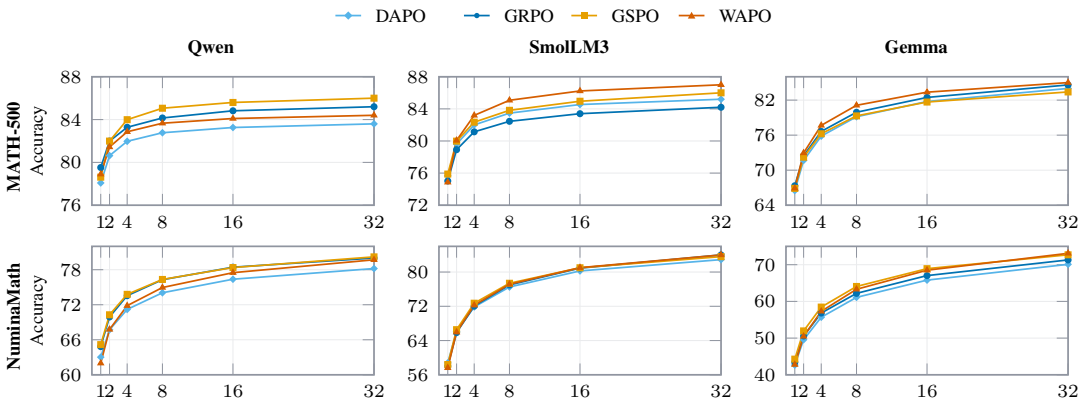


Figure 10: **MATH-500 and NuminaMath-LEAN pass@k curves.** The base model is omitted from this figure as its performance is substantially lower than the trained baselines, which would compress the y-axis range and make it harder to distinguish differences among the remaining curves.